Architectures à composants pour les systèmes embarqués communicants

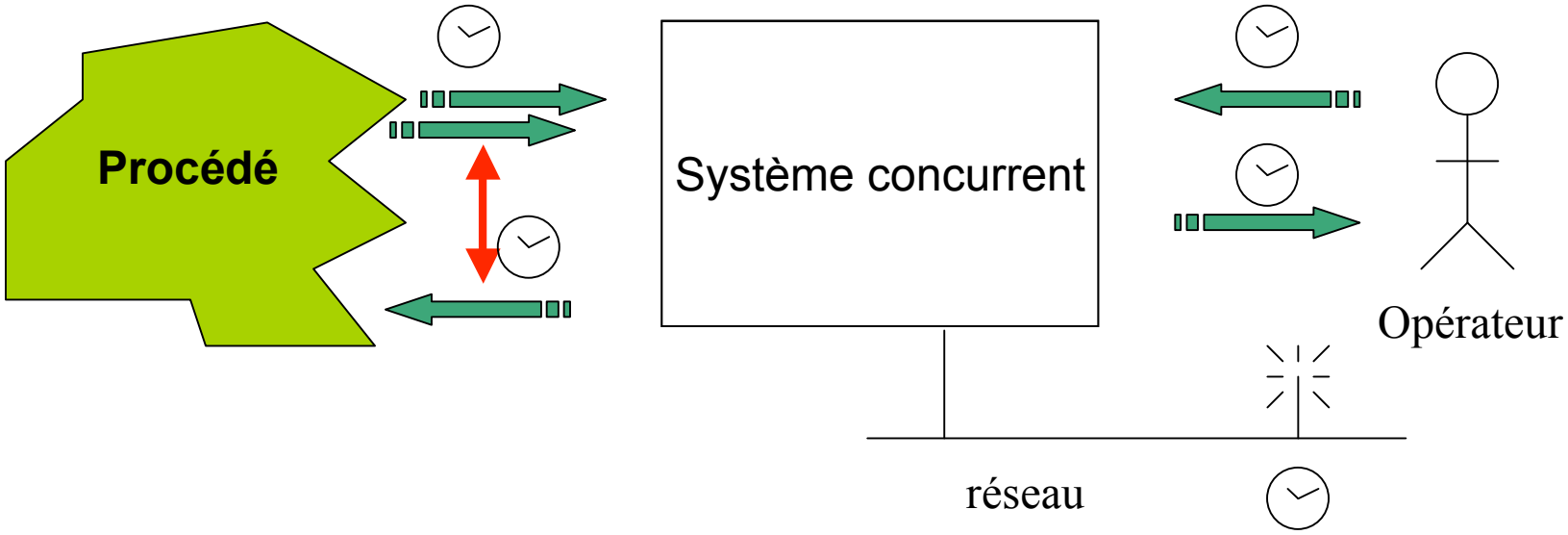
Julien De Antoni, Eyob Alemu, **Jean-Philippe Babau**,
Belgacem Ben Hédia, Fabrice Jumel, Jean-Charles Tournier

jean-philippe.babau@insa-lyon.fr

Plan

- Architectures pour les systèmes embarqués
- Formalisation et structuration des architectures
  - Composants
  - Modèles
  - Sémantique opérationnelle

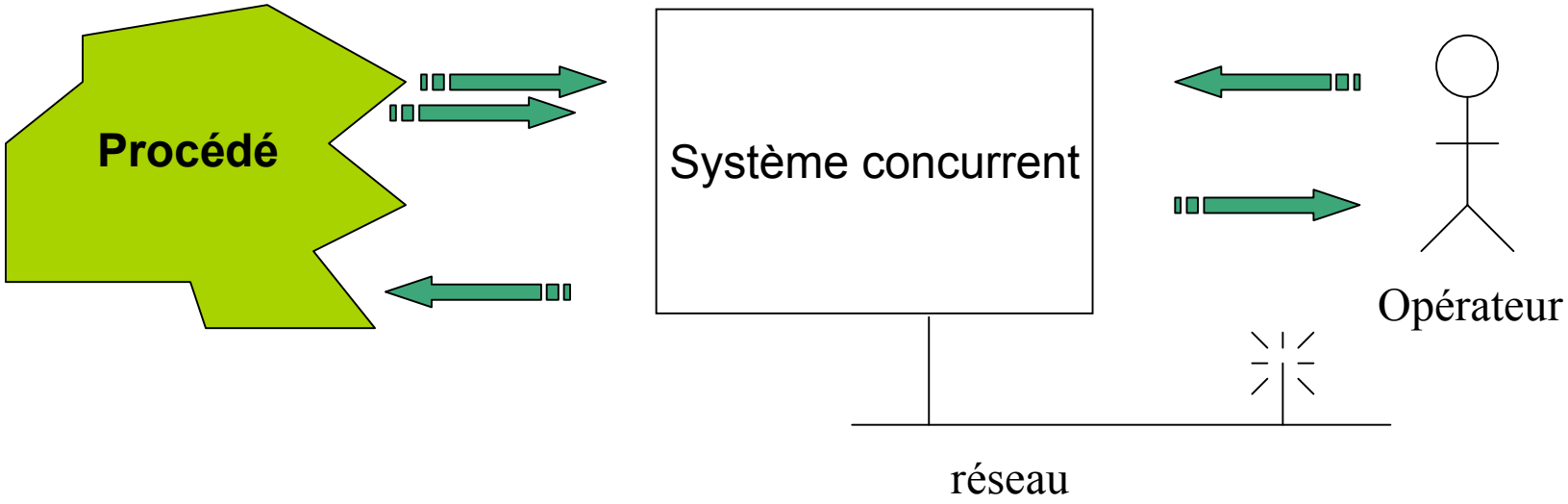
Les systèmes embarqués communicants



Contraintes temporelles

- acquisition, sorties, échéances

Les systèmes embarqués communicants



Contraintes temporelles

- acquisition, sorties, échéances

Contraintes d'embarquabilité

- Place limitée, contraintes physiques
- Contraintes d'énergie

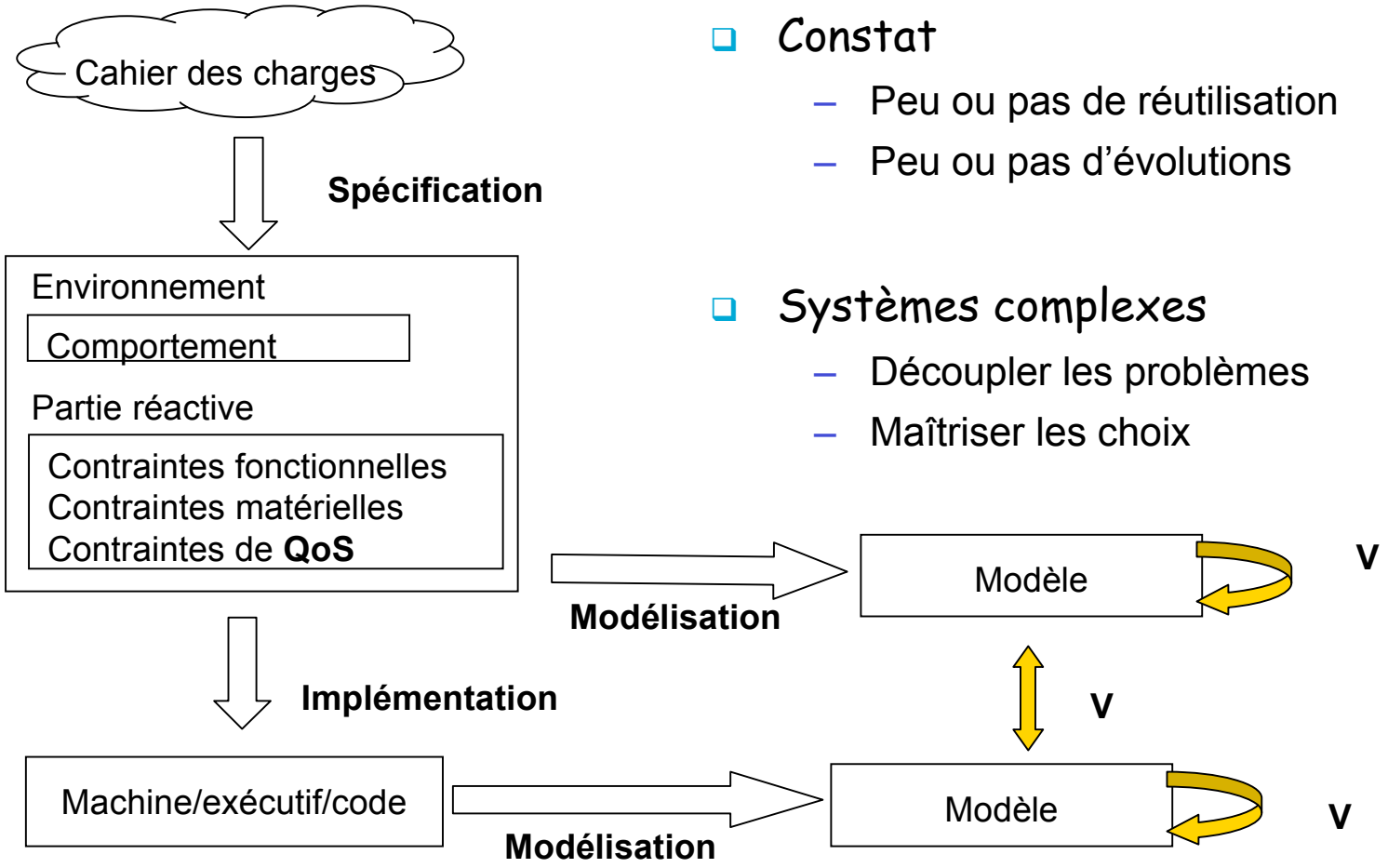
Contraintes de coût

Les spécificités des systèmes embarqués

- Systèmes fortement contraints
  - Contraintes temporelles
  - Contraintes d'embarquabilité
- Sûreté de fonctionnement
  - Systèmes critiques
  - Systèmes enfouis
- Développement
  - Ressources limitées et spécifiques
  - Systèmes prédictibles
  - Systèmes dédiés

Importance de l'implémentation pour respecter les contraintes

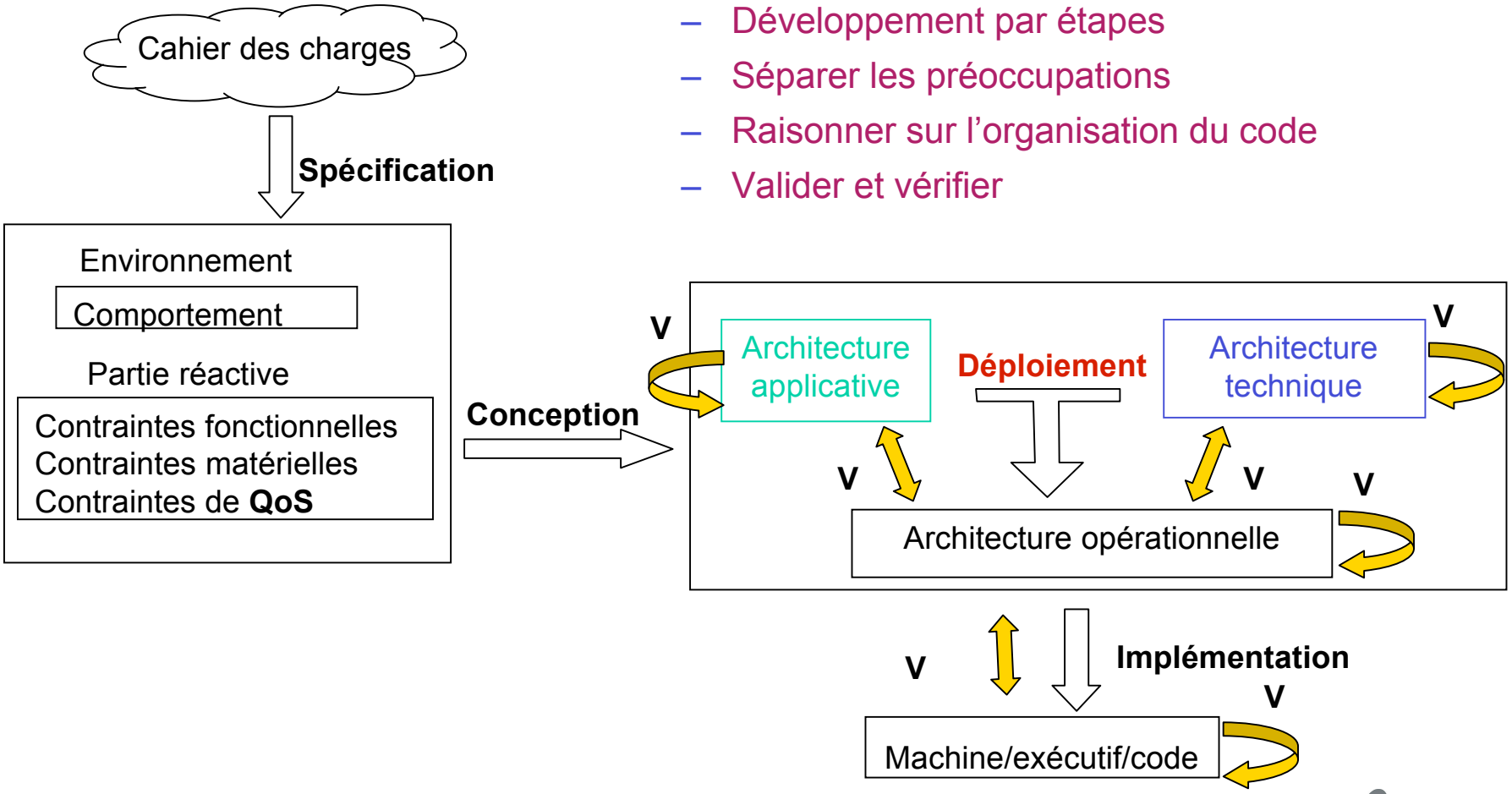
Développement



- Constat
  - Peu ou pas de réutilisation
  - Peu ou pas d'évolutions
- Systèmes complexes
  - Découpler les problèmes
  - Maîtriser les choix

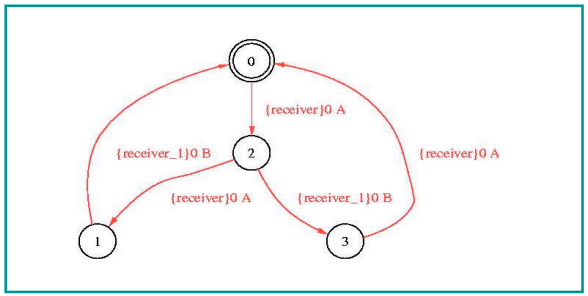
Améliorer le processus de développement

- Raisonner sur les architectures
  - Développement par étapes
  - Séparer les préoccupations
  - Raisonner sur l'organisation du code
  - Valider et vérifier



Les approches

- Maîtrise de la «qualité» des architectures
  - Langages de modélisation
    - ADL, UML 2, DSL
  - Paradigmes
    - Objet, **composant**
  - Model Driven Engineering
    - Modèles et méta-modèles



- Assurer la correction des architectures : analyse de performances
  - Approche analytique
    - Pire cas (RMA), bornes **[Min, max]**
  - Simulation exhaustive
    - Modèles à base d'automates temporisés (**IF/LTS**)
  - Simulation
    - Comportements complexes (**Matlab/Simulink**)

Les besoins

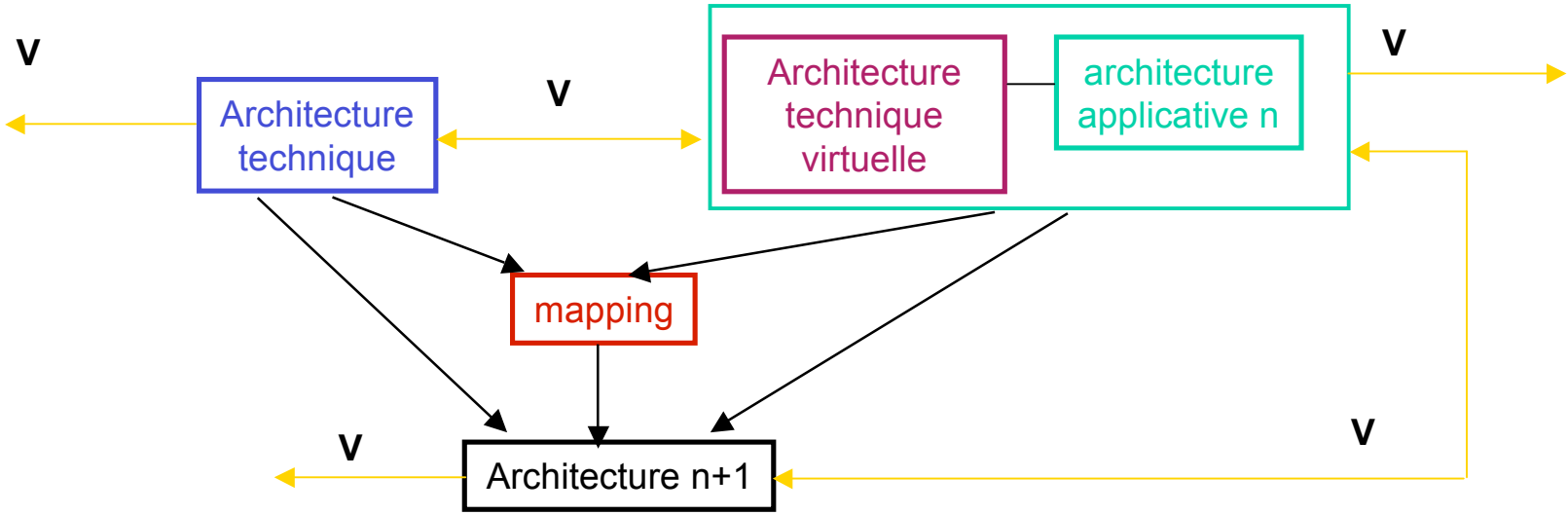
- Formalisation et structuration
  - Augmenter la qualité au sens du génie logiciel
  - Mise en œuvre de techniques formelles
- Comportements variables
  - Environnement
  - Applications
    - Durées
    - Ajout/retrait de modules
    - Modes de fonctionnement
  - Ressources
- Intégration de contraintes diverses
  - Temps réel / non temps réel
  - Temps, énergie, mémoire

Les besoins

- Portabilité des applications
  - Raisonner au niveau des architectures applicatives
- Evolution des applications et des plateformes
  - Dissocier les modèles d'architectures
- Intégration d'applications et de services
- Vérification avant déploiement

Phase de déploiement (raffinement)

- Approche «MDA» à base de composants
  - PIM ⊕ Plate-forme PM ⊕ mapping = PSM
  - Plateforme abstraite
  - Structuration à base de composants
  - sémantique IF, méta-modèles
  - Modélisation par étape



Les plateformes

- Un ensemble de ressources
  - Composant
  - Services d'accès
- Plateformes pour les systèmes embarqués
  - I/O (capteurs / actionneurs), IHM
  - Communication (réseau)
  - OS et HW (concurrence, mémoire, énergie)
- Importance de la QoS

Les Entrées / Sorties (I/O)

- Communication avec l'environnement
  - Bords de SA-RT : bords, acteurs de UML2
  - Procédé et environnement physique

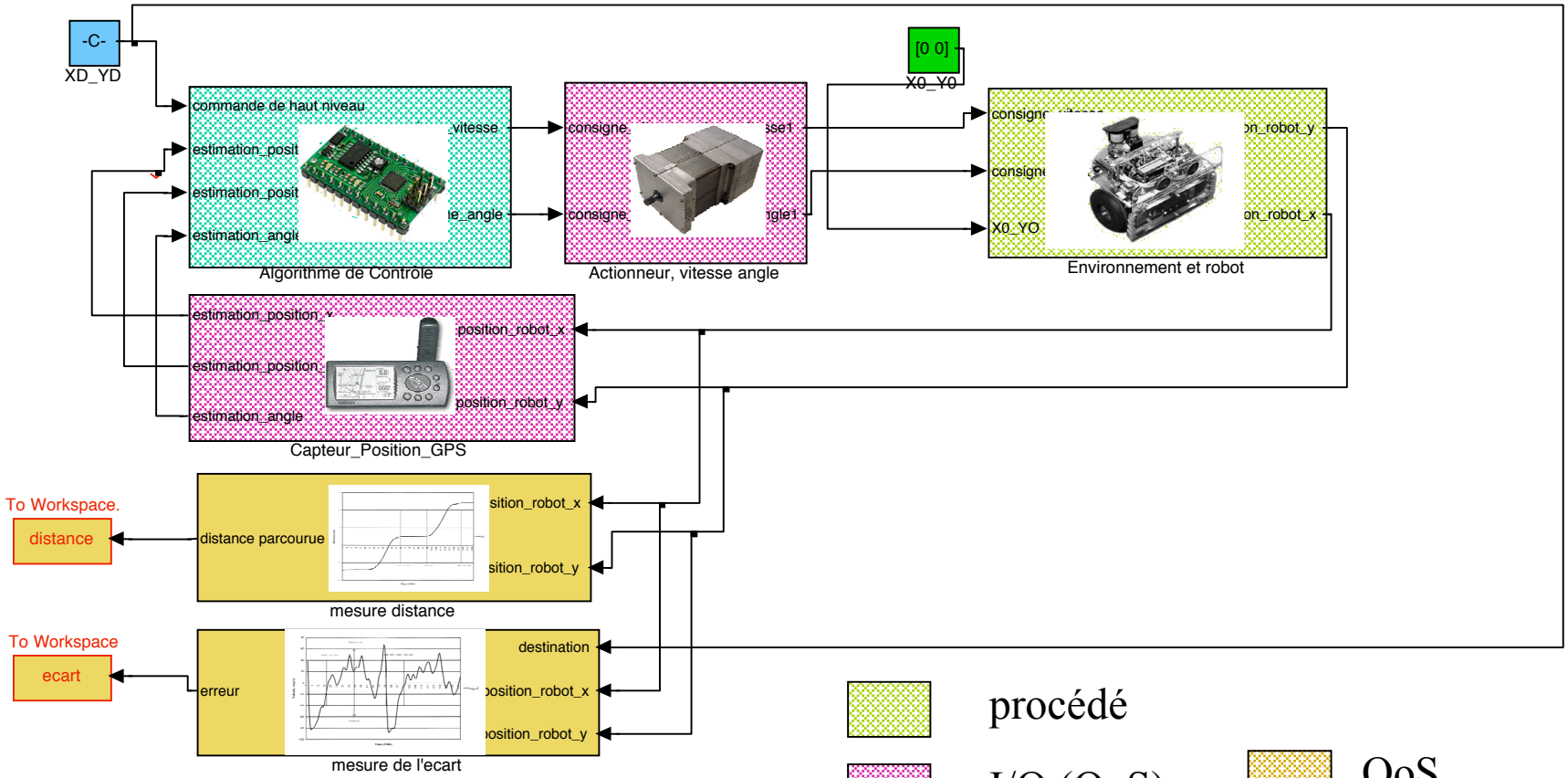
- Spécification
  - Les I/O sont vues à un haut niveau d'abstraction
  - Plateforme abstraite

- A la mise en œuvre
  - Capteurs / Actionneurs
  - Données de bas niveau

Les I/O et la QoS

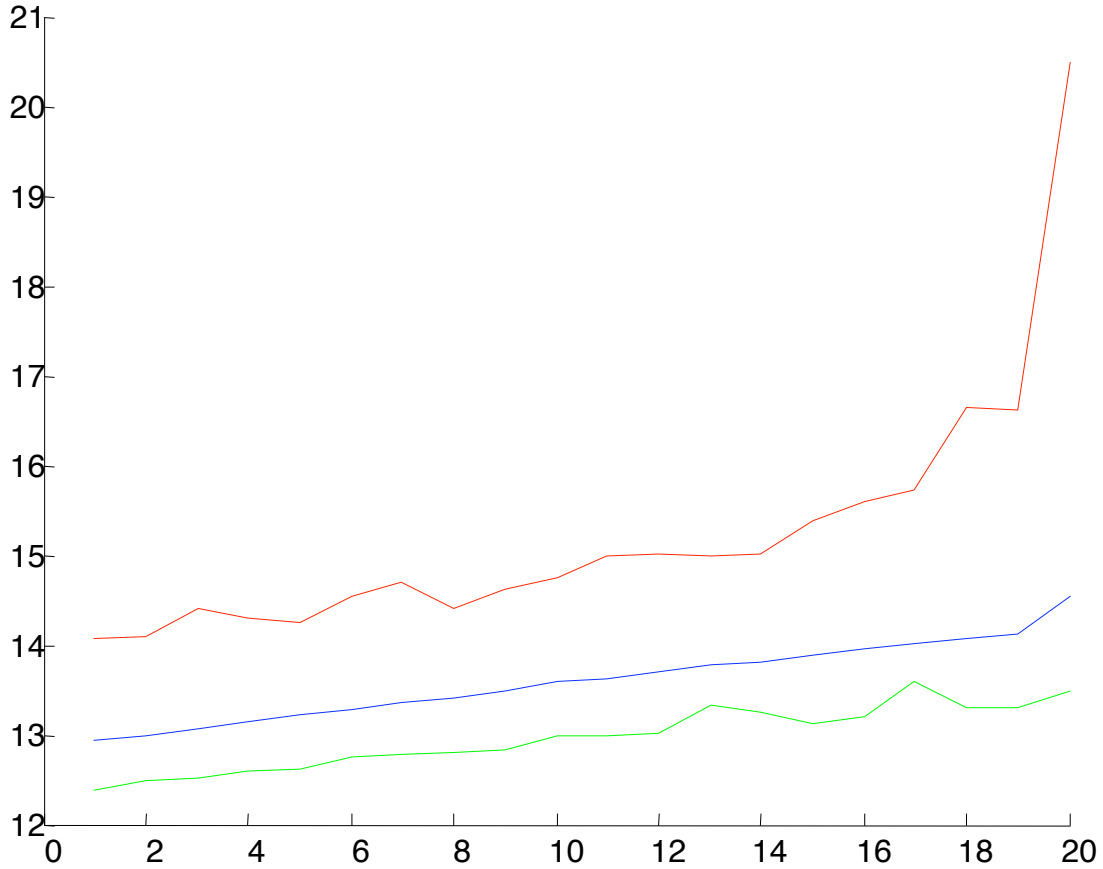
- Niveau applicatif
  - QoS applicative
  - QoS fournie à l'utilisateur
- Niveau «plateforme abstraite»
  - QoS sur les données en terme de fréquence, retard, précision
  - QoS fournie à l'application
- Dérivation de contrainte
  - Quelle QoS sur les I/O permet d'assurer une certaine QoS applicative
- Simulation Matlab / Simulink
  - QoS applicative = fonction(I/O QoS, comportement de l'applicatif)

Modèle Matlab



Résultats

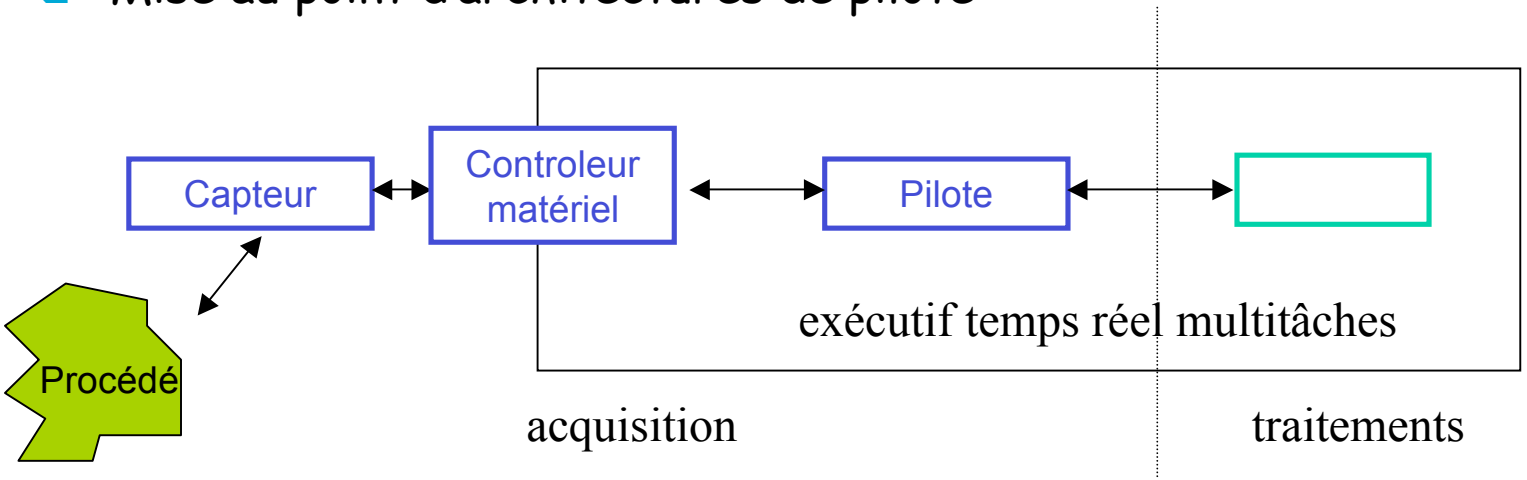
temps d'atteinte de l'arrivée



Période d'acquisition

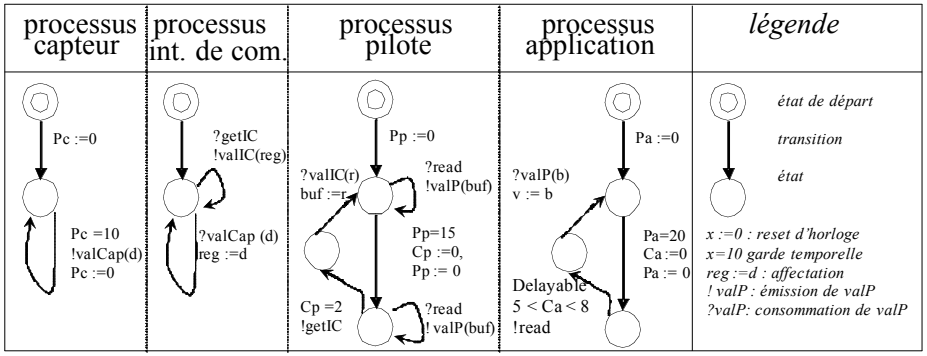
Système d'acquisition de données

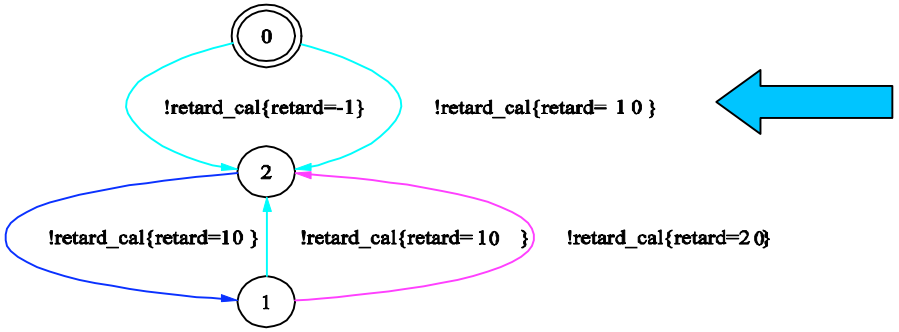
- Evaluation des plateformes techniques
- Réutilisation de services d'acquisition
  - Pilote de communication
- Qualité de service du pilote
  - QoS fournie en terme de fréquence, pertes, retards
- Mise au point d'architectures de pilote

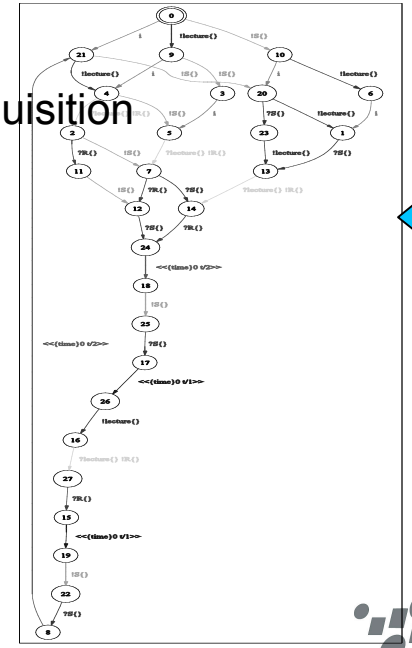


Modélisation

- Modèle IF
  - Flux de données
  - Compteurs d'occurrences
  - Observateurs
- Analyse
  - IF -> LTS -> LTS réduit
  - IF : modèle du code
  - LTS réduit : vue orientée QoS du composant d'acquisition

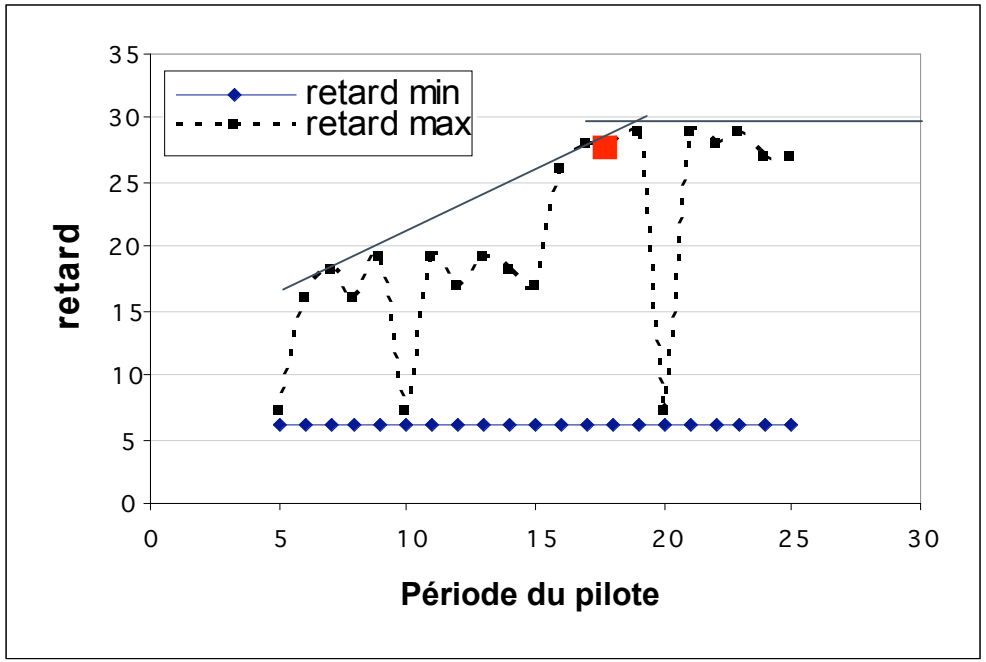






Système d'acquisition de données

- Principes architecturaux
  - Influence d'un paramètre

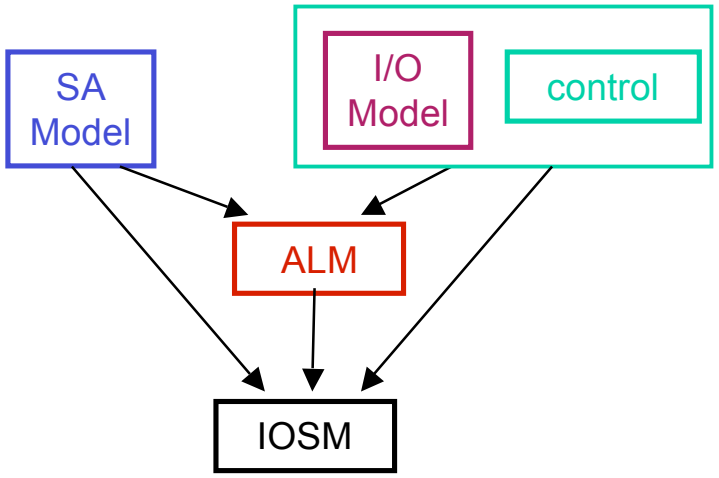


Temps de réponse du pilote : [1,2]
Temps de réponse de l'application : [6,7]
Période de l'application : 20
Période du capteur : 10

Analyse RMA

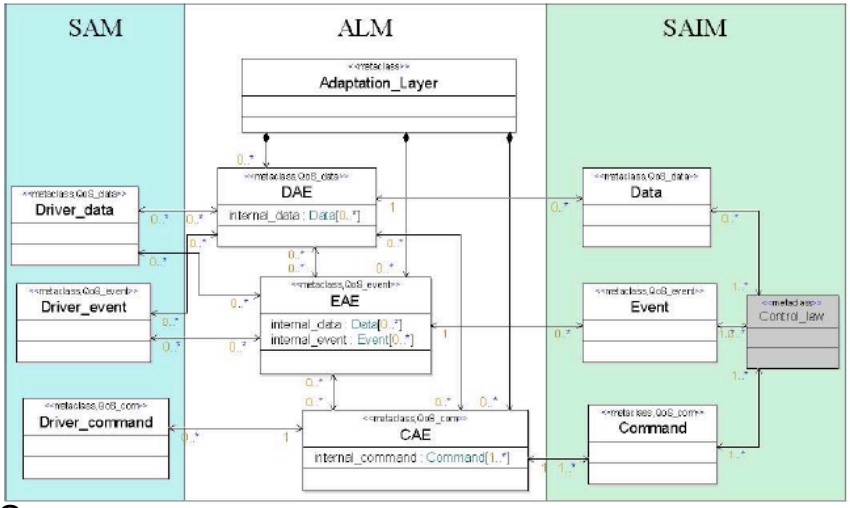
Architecture applicative SAIA
(Sensor Actuator Independant Architecture)

- Lien I/O <-> capteurs/actionneurs
  - QoS : fréquence, retards, corrélation, précision
- Indépendance vis-à-vis des mécanismes d'entrée/sortie
  - Simulation/prototypage
  - Portage sur autre cible
  - Modification (E/S, fonction)
- Architecture
  - Composants
  - Sémantique IF



Architecture SAIA

- 3 composants composites
  - SAIM
    - I/O de haut niveau : la QoS requise
  - SAM
    - I/O de bas niveau (drivers) : la QoS fournie
  - ALM
    - Connecteur / adaptateur
    - Assemblage de composants
    - Formatage, interprétation, adaptation de QoS



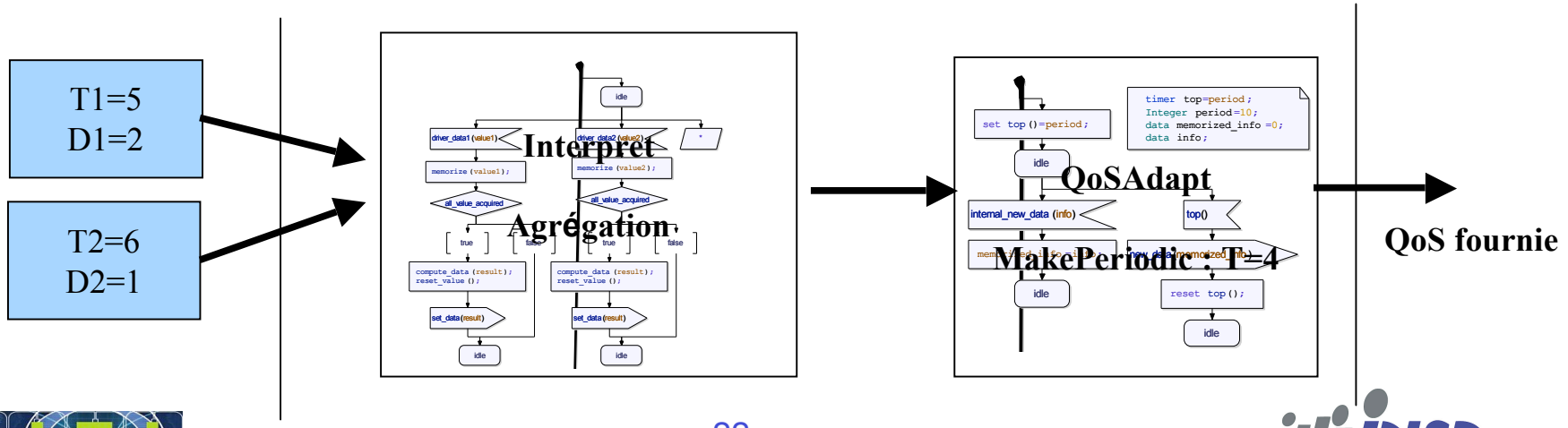
- Mise en oeuvre
  - Boîte à outils
  - Approche MIC (méta-modèles)

- Implémentation
  - Runner Up of the maRTian Task design competition
  - workshop ERTSI / RTSS'06 (Embedded Real Time System Implementation)

Analyse de la QoS fournie par l'ALM

- Approche analytique
  - Intervalles [min, max]
  - Analyse de type RMA

- Simulation exhaustive
  - IF
  - 3 272 (13 304) états et 5 829 (23 746) transitions sans (avec) observer IF
  - Réduction avec (sans) tick : 72 (19 états) / 83 (30) transitions

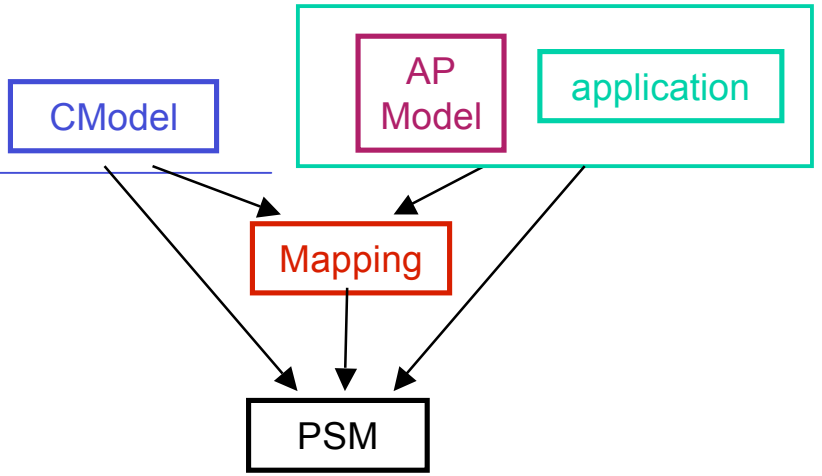


Contractualisation de la QoS

- QoS Fournie (ALM) « satisfait » QoS Requise (SAIM)
  - Intervalles : inclusion
  - LTS : la QoS requise *simule* la QoS fournie

- Modélisation de plateformes, de famille d'applications

Communications



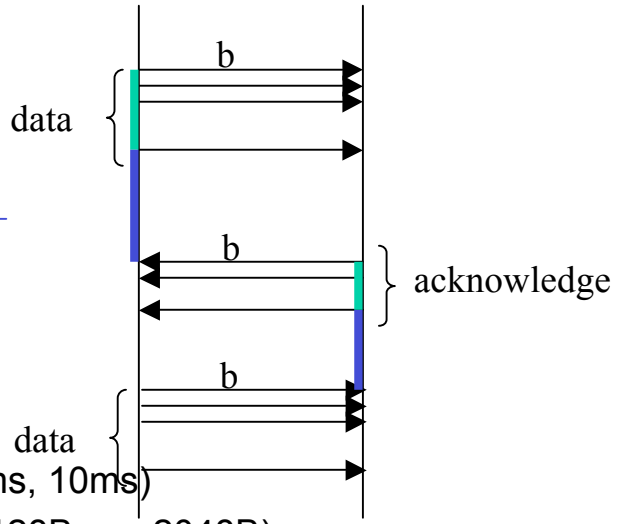
- Communication distante entre objets
  - A remoteSend to B

- «Abstract Platform»
  - Famille de protocoles « connexion oriented »
  - QoS applicative
    - Débit et taille maximale des données (MTU) (application / crédits)
    - Retards et pertes
    - Niveau de garantie : Garanti / Contrôlé / Best-Effort

- «Concrete Platform»
  - Sans fil pour l'embarqué
  - Irda, Bluetooth

- « Mapping »
  - Lien fonctionnel
  - QoS

IrDA (half-duplex)



- Paramètres de QoS
  - Br : Baud rate (9.600 kbps, 11.52 kbps, …, 16 Mbps)
  - Link disconnect (3s, 8s, 12s, … 40s)
  - **MTt** (50ms,100ms, 250ms, 500ms), **mtt** (0ms, 0.01ms, …, 5ms, 10ms)
  - Data Size : Window Size (1-127 frames) x Frame Size (64B, 128B, …, 2048B)
  - Additional BOF
- Débit $\frac{Br \cdot MTt}{MTt + 2 \cdot mtt + AckSize/Br} > \text{Minimum(pr,br)}$
- Retard
  - Pas de prise en compte des retards logiciels, délais de propagation négligeables, pas de fautes
  - Délai de transmission : $\frac{MTU}{Br}$
  - Surcharge due aux acknowledge Ao : $\left\lceil \frac{MTU}{DS} \right\rceil \cdot \left(2 \cdot mtt + \frac{AckSize}{Br} \right)$
    - si MTU = DS : Ao = 0

QoS Mapping

- Dérivation de contraintes
  - $Throughput_{Req}(br,pr) <= Throughput_{Prov}(Br,MTt,mtt,AckSize)$
  - $Latency_{Req}(delay) <= Latency_{Prov}$ (Br, **MTU**, DS*, mtt, AckSize*)

- Négociation
  - Définition des paramètres PIM
    - *Garanti* : une solution
    - *Contrôlé* : différentes solutions possibles
  - Dérivation de contraintes pour IrdA
    - Plusieurs solutions possibles
  - *Garanti / Contrôlé* : négociation jusqu'à trouver une solution acceptable
  - *Best effort*
    - Envoi de paramètres
    - Réception de paramètres du receveur
    - Réévaluation des paramètres PIM

Une architecture opérationnelle à base de composants pour la gestion de la qualité de service

Une architecture opérationnelle pour la gestion de la qualité de service

- Systèmes embarqués communicants
  - Gestion sure des ressources
  - Ajout / retrait de composants
  - Adaptation (applications, ressources)

- Composants
  - Services fournis et requis
  - Modèle de composants Fractal de France Télécom R&D
  - Think : implémentation de Fractal pour l'embarqué
  - Composition : contrats
    - Signature / pré-post / protocole / **Qualité de Service**

Composants, QoS et architectures

- Principes
  - Composants gérés à l'exécution
  - Séparation des préoccupations
  - Gestion de contrats : établissement, suivi et adaptation
  - Container
  - Contrôle des ressources

- Composants pour les systèmes distribués [EJB, OSGI, CCM, COM, .NET]
  - Pas de contrôle des ressources
  - Pas de gestion de la QoS liée aux ressources

- Composants pour l'embarqué [PEC99][OLK00][SAH01][UVH 01][IN 02]
  - Pas de composants à l'exécution (hormis Think [FSL 02])
  - Gestion a priori (compilation) de certaines contraintes

Les principes de Qinna

- Composant
  - Tout est composant
  - Gestion dynamique à l'exécution (Think)
  - Séparation des préoccupations
    - F-Component, QoSComponent, QoSBroker, QoSManager, QoSDomain, QoSObserver

- QoS
  - Contrats : enveloppes de QoS
  - Gestion dynamique : tout changement entraîne une recherche de solution
  - Généricité : types abstraits

Architecture Qinna

- Composants de gestion de la QoS
  - Service : interaction d'un ensemble de composants (F-Component)
    - 4 niveaux (application, services, OS, ressources)
  - Un QoSComponent : initialisation et vérification dynamique des contrats
    - Un par F-Component
  - Un QoSManager : vue orientée QoS du F-component
    - Un par QoSComponent
    - Un contrat fourni s'appuie sur n sous-contrats requis (table de mapping)
  - Un QoSbroker : test d'admission
    - Un pour l'ensemble des QoSComponent de même type
    - 1 seul (CPU, mémoire, …) ou n (mémoire) QoSComponent
  - Un QoSObserver : suivi des contrats
    - Un pour l'ensemble des QoSComponent
  - Un QoSDomain : politiques d'adaptation et d'observation
    - Point d'entrée unique de l'architecture

Gestion dynamique de la QoS

- Contrat
  - Discrétisation des niveaux
    - *pas* de maintenabilité
  - Compromis granularité / nombre d'opérations
  - Hétérogénéité : default, unreliable, translate

- Suivi des contrats
  - Changement
    - Demande / Arrêt d'un F-component
    - Modification : demande d'une application ou état d'une ressource
  - Recherche d'un contrat acceptable
    - Parcours de l'arbre défini par les QoSManager
    - Test d'admission par appel des QoSBrokers

- Bilan
  - Compromis gaspillage de QoS / adaptations
  - Confiance : Contrôle de ce qui est alloué par le QoSComponent : niveau ressources
  - Impact de la période d'observation en terme de retards

Evaluation de Qinna

- Etudes de cas
  - Ordonnancement
    - Structures hiérarchiques, régisseur
  - Pic de demande, de charge
    - « saut(s) » selon la forme du pic, suivi périodique
  - Variation d'une ressource
    - « Marches » prédéfinies, suivi événementiel
  - Application vidéo
    - 7 QoSComponent, processeur 200Mhz, 60 Mo de RAM
    - Établissement et adaptation d'un contrat : 4,38 ms et 1,08 ms
    - Place : 749 ko, 1,5 %

- Utilisation
  - Systèmes multimédias
    - Adaptation, coûts raisonnables
  - Systèmes temps réel
    - Structuration et intégration des politiques de gestion de la QoS

Qinna

- ARA de l'ANR : REVE (CEA,INRIA Futur, CITI, INRETS)
  - Architecture à composants pour l'adaptation aux changements de contexte
  - Contexte
    - Etat des ressources
    - Plateforme d'exécution
  - Application à un *viewer* d'images distantes

Conclusion

- Architectures à base de composants «QoS aware» pour les Systèmes embarqués communicants

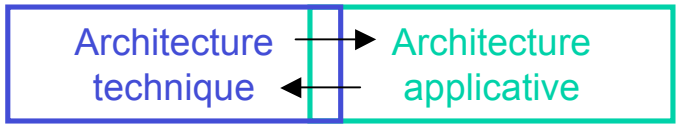
- Prise en compte des entrées / sorties et des communications
  - Composants « hétérogènes »
  - Plateformes abstraites et concrètes
  - Lien plate-forme abstraite et plate-forme concrète
    - Connecteur (adaptation)
    - Négociation
  - QoS
    - Dérivation de contraintes
      - application -> I/O, APM ->CPM
    - Composition et connexion (F/R)

- Gestion dynamique de la QoS
  - Gestion dynamique de contrats de QoS
    - Dérivation de contrainte, test d'admission, négociation
  - Discrétisation, enveloppe, compromis

Perspectives

- Structuration et formalisation
  - Composants et composition
    - assemblage, contractualisation, interdépendance
  - Caractérisation d'assemblages à l'aide de théories

- Hétérogénéité
  - Paradigmes et langages
    - Matériel / logiciel
  - Sémantiques
    - Synchrone / asynchrone, discret / continu



Perspectives

- Distribution
  - Contraintes stochastiques
  - Aspects probabilistes
- Implémentation
  - Génération automatique
  - Optimisation
  - Ordonnancement

des questions !?!?

Discussion